Updated Tagging Design

Goal

Provide an updated tagging API that adds indexed string capabilities, arbitrary JSON with associations, and the ability to tag individual collection revisions.

Traceability

CMR-1952 - JIRA project doesn't exist or you don't have permission to view it.

Tagging Epic (Note, stories for the capabilities in this review have not yet been created.)

Background

Tagging was first implemented in early Fall of 2015 based on requirements that had been defined to replace GCMD tags. After the tagging implementation was complete some new use cases emerged for other clients and a few things that were still needed for GCMD.

Earthdata Search Client (EDSC)

EDSC has a few different types of data that it associates with collections. The EDSC expressed an interest in using CMR's tags to allow storing this information in the CMR. They included things like the following.

- GIBS configuration, or, at the very least, the fact that a collection has GIBS visualizations
- OPeNDAP configuration, or, at the very least, the fact that a collection has OPeNDAP access
- Whether a collection is in MODAPS
- Whether a collection is featured

The current tag capabilities would work for tagging if a collection is in MODAPS and featured collections but wouldn't be able to handle the use cases that involves associating more complicated data with a tag. Additionally the developers noted that the current split between tag namespace and value isn't necessary. It would be simpler to identify a tag with a single string.

GCMD Curation

GCMD needs the ability to curate individual collection revisions. Each revision could go through several levels of review status:

- Accepted for review
- In review
- Review-accepted
- Review-disputed (rejected)

They may want to store additional review comments and other data along with the review status.

Additionally they may need to tag collections associated with a portal. "Tagging helps us overcome shortcomings in a ridged data structure and allow these data providers to relate metadata in ways they find make since. A portal is a subset or grouping of records which have some logical relationship and tags helps us capture that relationship."

The current tag capabilities does not allow tagging individual revisions. Also the ability to store additional review comments and other data would not be possible in the current model.

CWIC

CWIC needs to be able to track a few different kinds of values with collections

- CWIC Status - "prod" or "test"
- Data Provider - "NASA", "ISRO", etc.
- "Native Id" - This is the CWIC identifier for a collection and is not in the collection metadata. CWIC needs to be able to search for collections by this id.

The current tag capabilities would not allow assigning arbitrary strings to each collection. It could be done with separate tags per collection but this would create a lot of tags in the CMR.

Changes to Current Design

This is a high level list of changes to the current design.

- Tags
  - Namespace and value would be combined and called "tag-key". Example tag-key: "gov.nasa.earthdata.search.gibs_configuration"
  - The following fields would be removed. They were added with the original requirements but I couldn't see a need for them with the given use cases. If any of these are still needed we could continue to maintain them
    - category
- Tag Associations
  - When tags are associated with a collection the association can contain arbitrary JSON data. If the data is a string it will be indexed and searchable.

Client Use Case Walkthroughs

This section walks through using the new API (detailed below) to satisfy the different client use cases.

EDSC

Tagging a Collection with GIBS Configuration

This POST sends a list of tag associations to make. There are two different collections which are tagged with GIBS configuration information. The value of "data" can be any arbitrary JSON up to 32KB.

```
POST /search/tags/gov.nasa.earthdata.search.gibs_configuration/associations
[{"concept-id": "C2-PROV1", "data": {...}},
 {"concept-id": "C3-PROV1", "data": {...}}]
```

The GIBS configuration can be retrieved with collection JSON search results using the include_tags parameter. Each collection will contains the tag-key mapped to the data it was associated with.

```
GET /search/collections.json?include_tags=gov.nasa.earthdata.search.*
Response:
{
  "feed" : {
    "entry" : [ {
      ...
      "tags" : {"gov.nasa.earthdata.search.gibs_configuration":
{"associationDate":"2015-01-01T00:00:00.0Z",
                                                                  "data": {...}}},
      ...
    } ]
  }
}
```

Tagging a Collection as in MODAPS

Associate the collections in MODAPS with the tag. The "data" key is always required. When associating collections without any real data the value of true should be used.

```
POST /search/tags/gov.nasa.earthdata.search.in_modaps/associations
[{"concept-id": C1-PROV1", "data":true},
 {"concept-id": C2-PROV1", "data":true}]
```

When the tag is returned in collections results the map indicates if that tag is present. A collection without the tag would not have the tag-key in the tag-key map.

```
GET /search/collections.json?include_tags=gov.nasa.earthdata.search.*
Response:
{
  "feed" : {
    "entry" : [ {
      ...
      "tags" : {"gov.nasa.earthdata.search.in_modaps":
{"associationDate":"2015-01-01T00:00:00.0Z"}},
      ...
    } ]
  }
}
```

GCMD Curation

A review is started on a collection. A single revision of the collection is tagged to indicate it is in review.

```
POST /search/tags/gov.nasa.gcmd.review_status/associations
[{"concept-id": "C1-PROV1",
  "revision-id": 4,
  "data": "IN_REVIEW"}]
```

The collection undergoes a review and the GCMD reviewer adds some information using a review_notes tag. Note that this is just some sample data. The actual data can be whatever best suits GCMD.

```
POST /search/tags/gov.nasa.gcmd.review_notes/associations
[{"concept-id": "C1-PROV1",
  "revision-id": 4,
  "data": {"overview": "Many misspellings. These should be corrected",
           "score": 45}}]
```

The collection revision review status is updated to indicate the revision is rejected.

```
POST /search/tags/gov.nasa.gcmd.review_status/associations
[{"concept-id": "C1-PROV1",
  "revision-id": 4,
  "data": "REVIEW_DISPUTED"}]
```

The GCMD client or other clients can retrieve the review information.

```
GET
/search/collections.json?concept_id=C1-PROV1&revision_id=4&all_revisions=true&include_tags=gov
.nasa.gcmd.*
Response:
{
  "feed" : {
    "entry" : [ {
      ...
      "tags" : {"gov.nasa.gcmd.review_status": {"associationDate":"2015-01-01T00:00:00.0Z",
                                                "data": "REVIEW_DISPUTED"},
                "gov.nasa.gcmd.review_notes": {"associationDate":"2015-01-01T00:00:00.0Z",
                                               "data": {"overview": "Many misspellings. These
should be corrected",
                                                        "score": 45}}},
      ...
    } ]
  }
}
```

Find collection revisions that are IN_REVIEW.

```
POST -H "Content-Type: application/json" /search/collections?all_revisions=true -d
'{"condition": {"tag": {"tag_key": "gov.nasa.gcmd.review_status", "tag_data": "IN_REVIEW"}}}'
```

CWIC

Associate a collection with the CWIC Native Id (Note that this is different than the native id in the CMR which is an internal name for a different identifier). The CWIC data provider and CWIC status would all be stored in the CMR in a similar way.

```
POST /search/tags/org.ceos.wgiss.cwic.native_id/associations
[{"concept-id": "C1-PROV1",
  "data": "Global Maps of Atmospheric Nitrogen Deposition, 1860, 1993, and 2050"}]
```

Search for collections by CWIC Native Id. It must be URL encoded.

```
GET
/search/collections?tag_data[org.ceos.wgiss.cwic.native_id]=Global%20Maps%20of%20Atmospheric%2
0Nitrogen%20Deposition%2C%201860%2C%201993%2C%20and%202050
```

Retrieve all CWIC tags with collection search results.

```

```

```
GET /search/collections.json?include_tags=org.ceos.wgiss.cwic.*
Response:
{
  "feed" : {
    "entry" : [ {
      ...
      "tags" : {"org.ceos.wgiss.cwic.native_id": {"associationDate":"2015-01-01T00:00:00.0Z",
                                                  "data": "Global Maps of Atmospheric Nitrogen
Deposition, 1860, 1993, and 2050"},
                "org.ceos.wgiss.cwic.data_provider":
{"associationDate":"2015-01-01T00:00:00.0Z",
                                                    "data": "NASA"},
                "org.ceos.wgiss.cwic.cwic_status":
{"associationDate":"2015-01-01T00:00:00.0Z",
                                                  "data": "prod"}},
      ...
    } ]
  }
}
```

Remove associations from a collection. The body of the request is a JSON array of CMR concept ids

```
DELETE /search/tags/org.ceos.wgiss.cwic.native_id/associations
["C1-PROV1]
```

Tag API Overview

- /search/tags
  - GET - Find tags by combinations of parameters
    - Supported parameters: tag-key prefix, originator-id,
  - /:tag-key
    - PUT - create/update a tag
      - Not necessary to create the tag. Adding associations will automatically create the tag.
    - GET - retrieve a tag.
    - DELETE - remove a tag
    - /associations
      - POST - add multiple new associations optionally with data
        - Body is JSON array of maps per concept to associate. Each map can have:
          - concept-id - id of concept to tag
          - revision-id - Optional to tag a specific revision
          - data - This can be one of two types
            - A string will be indexed and searchable.
            - Other JSON data can be used up to 32 K. The data is retrievable with collections results.
      - DELETE - Remove associations from a collection
        - Body is a JSON array of concept ids.

HTTP Status Codes

The following table explains what HTTP Status codes in responses indicate.

HTTP Status Code	Description	Usage
200	Success	This is used when performing a non modifying action is taken to retrieve data or when a record has been updated or deleted successfully.
201	Creation	This is used to indicate a new record has been created.
400	Bad request	The format of the request was invalid
401	Unauthorized	Your user did not have permission to take that action.
404	Not Found	This would be returned when attempting to retrieve, update, or delete an item that does not exist
422	Invalid	The entity sent during a request was not valid. For example this would be sent if trying to create a tag association without specifying a concept id.

500	Internal Error	An internal error in the CMR has occurred

Tag Keys are case insensitive

Tag keys will be stored and indexed in lower case. They are never case sensitive. A mix of cases can be used on any tag API including creating tags, associating tags with collections, and searching for tags and with tag keys. Any input tag keys will automatically be converted to lower case. Tag keys in search responses will be shown in lower case as well.

Searching for collections by tag

The parameter tag_key will be supported as a string search param:

```
curl -g "https://cmr.earthdata.nasa.gov/search/collections?tag_key=gov.nasa.earthdata.search.*"
```

The tag_key string is treated as a pattern by default. An explicit tag with asterisks and questions marks can be searched by setting the param option pattern to false. This is different than the typical CMR search parameter behavior because it is assumed that clients will be searching by pattern most of the time.

Similarly, the tag_key parameter will be supported as a JSON Query condition:

```
curl -XPOST -H "Content-Type: application/json" https://cmr.earthdata.nasa.gov/search/collections -d
'{"condition": {"tag": {"tag_key": "gov.nasa.earthdata.search.*"}}}'
```

Search for collection by tag indexed string

If a string is associated with the collection the string will be indexed and searchable through the query parameter and JSON Query API

For the query parameter API, the parameter used is `tag_data` in the format of

```
tag_data[<tag-key>]=<search-string>
```

Example:

```
curl -g
"https://cmr.earthdata.nasa.gov/search/collections?tag_data[org.ceos.wgiss.cwic.native_id]=some_cwic_id_1
```

JSON Query Example

```
curl -XPOST -H "Content-Type: application/json" https://cmr.earthdata.nasa.gov/search/collections -d
'{"condition": {"tag": {"tag_key": "org.ceos.wgiss.cwic.native_id", "tag_data": "some_cwic_id_1"}}}'
```

Search for collection by tag originator id

Collections can be found using the user id of the original user to create the tag.

Example:

```
curl -g "https://cmr.earthdata.nasa.gov/search/collections?tag_originator_id=jgilman
```

JSON Query Example

```
curl -XPOST -H "Content-Type: application/json" https://cmr.earthdata.nasa.gov/search/collections -d
'{"condition": {"tag": {"originator_id": "jgilman"}}}'
```

Retrieving tags with collection search results

Tags can be retrieved with collection search results. The tags will be returned with each collection search result with tag data corresponding to one of the following two types:

- string - Indicates the tag is associated with the collection with an indexed string value.
- Other JSON Data - Indicates the tag is associated with the collection with arbitrary JSON data.

The association date (i.e. revision date of the tag association information) will be returned as well. This gives the date the collection was last associated with this tag.

JSON Search Response

```
curl -g
"https://cmr.earthdata.nasa.gov/search/collections.json?include_tags=gov.nasa.earthdata.search
.*
Response:
{
  "feed" : {
    "entry" : [ {
      // ...
      // Tags is a map of tag-key to tag association information
      "tags" : {"gov.nasa.earthdata.search.foo": {"associationDate":"2015-01-01T00:00:00.0Z"},

                "gov.nasa.earthdata.search.bar": {"associationDate":"2015-01-01T00:00:00.0Z",
                                                  // a tag associated with a string
                                                  "data": "something"},
                "gov.nasa.earthdata.search.chew": {"associationDate":"2015-01-01T00:00:00.0Z",
                                                   // a tag with arbitrary JSON data
                                                   "data": {...}}},
      ...
    } ]
  }
}
```

Atom Search Response

Note that if arbitrary JSON data is associated with a tag it will be returned as JSON embedded in the XML and properly escaped. The CMR will not attempt to parse or convert the JSON to XML.

```
curl -g
"https://cmr.earthdata.nasa.gov/search/collections.atom?include_tags=gov.nasa.earthdata.search
.*
Response:
<feed>
  ...
  <entry>
    <id>C179003030-ORNL_DAAC</id>
    <title type="text">15 Minute Stream Flow Data: USGS (FIFE)</title>
    ...
    <echo:tags>
      <echo:tag associationDate="2015-01-01T00:00:00.0Z">
        <echo:tagKey>gov.nasa.earthdata.search.foo</echo:tagKey>
      </echo:tag>
      <echo:tag associationDate="2015-01-01T00:00:00.0Z">
        <echo:tagKey>gov.nasa.earthdata.search.bar</echo:tagKey>
        <echo:data>something</echo:data>
      </echo:tag>
      <echo:tag associationDate="2015-01-01T00:00:00.0Z">
        <echo:tagKey>gov.nasa.earthdata.search.chew</echo:tagKey>
        <echo:data>{...}</echo:data>
      </echo:tag>
    </echo:tags>
  </entry>
</feed>
```

Metadata Search Response

This shows how the tags associated with a particular collection are returned with the metadata search results. Note that if arbitrary JSON data is associated with a tag it will be returned as JSON embedded in the XML and properly escaped. The CMR will not attempt to parse or convert the JSON to XML.

```
curl -g
"https://cmr.earthdata.nasa.gov/search/collections.dif?include_tags=gov.nasa.earthdata.search.
*
Response:
<results>
  ...
  <result concept-id="C179003030-ORNL_DAAC" format="application/dif+xml" revision-id="17">
    <DIF>
      ...
    </DIF>
    <tags>
      <tag associationDate="2015-01-01T00:00:00.0Z">
        <tagKey>gov.nasa.earthdata.search.foo</tagKey>
      </tag>
      <tag associationDate="2015-01-01T00:00:00.0Z">
        <tagKey>gov.nasa.earthdata.search.bar</tagKey>
        <data>something</data>
      </tag>
      <tag associationDate="2015-01-01T00:00:00.0Z">
        <tagKey>gov.nasa.earthdata.search.chew</tagKey>
        <data>{...}</data>
      </tag>
    </tags>
  </result>
</results>
```

XML Reference Search Response

This shows how the tags associated with a particular collection are returned with the XML reference search results. Note that if arbitrary JSON data is associated with a tag it will be returned as JSON embedded in the XML and properly escaped. The CMR will not attempt to parse or convert the JSON to XML.

```
curl -g
"https://cmr.earthdata.nasa.gov/search/collections.xml?include_tags=gov.nasa.earthdata.search.
*
Response:
<results>
  ...
  <references>
    <reference>
      <name>15 Minute Stream Flow Data: USGS (FIFE)</name>
      <id>C179003030-ORNL_DAAC</id>

<location>https://cmr.earthdata.nasa.gov:443/search/concepts/C179003030-ORNL_DAAC/17</location
>
      <revision-id>17</revision-id>
      <tags>
        <tag associationDate="2015-01-01T00:00:00.0Z">
          <tagKey>gov.nasa.earthdata.search.foo</tagKey>
        </tag>
        <tag associationDate="2015-01-01T00:00:00.0Z">
          <tagKey>gov.nasa.earthdata.search.bar</tagKey>
          <data>something</data>
        </tag>
        <tag associationDate="2015-01-01T00:00:00.0Z">
          <tagKey>gov.nasa.earthdata.search.chew</tagKey>
          <data>{...}</data>
        </tag>
      </tags>
    </reference>
    ...
  </references>
</results>
```

Questions/Assumptions

- Are there use cases that this misses?
  - Answer from review: Any additional use cases were captured in the review and include in the design above.

- Do you need to associate tags with a query that finds collections?
    - This would create the tag equivalent to a "smart folder" where every time the collections associated with a tag are retrieved the query will be evaluated.
    - This is more difficult to implement but is possible in the current design. We'd rather not have to address that in the current set of work.
    - Answer from review: GCMD and others like this feature but do not need it yet.
- **Assumption for transition of current data is that we will remove all existing tags and no existing data will be maintained.**
    - Clients will need to recreate existing tags with the addition of the new API.
    - The new tag design is a large departure from the current implementation. The current implementation has not been around long. It's assumed that clients have only been using this in a testing capacity.
    - We can transition the data but this is going to be more work.
    - Answer from review: Both GCMD and EDSC agree that this is ok.